Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

TRABAJO FIN DE GRADO

Monitorización del juego Travian

Jorge Vergés Montano

Monitorización del juego Travian

AUTOR: Jorge Vergés Montano
TUTOR: Eduardo Boemo Scalvinoni

Escuela Politécnica Superior
Universidad Autónoma de Madrid
Junio de 2019

Jorge Verges Montano.
Monotorización del juego Travian

IMPRESO EN ESPAÑA – PRINTED IN SPAIN

Resumen (castellano)

En esta memoria se expone el camino realizado para conseguir implementar un sistema de monitorización con almacenamiento de datos y una herramienta de análisis de éstos, cuyo objetivo es intentar discernir la posible inactividad de un jugador.

Es un proyecto multidisciplinar, ya que implementa todos los conceptos que necesita cualquier aplicación en cualquiera de sus capas, desde la capa básica hardware pasando por las capas de red, como la configuración de un sistema operativo hasta el propio desarrollo software.

En este trabajo se describe la situación actual del contexto del proyecto en la sección de estado del arte, en la que también se describen las herramientas empleadas y los usos dados en este TFG.

Existe otra sección llamada diseño en la que se explica de forma detallada las decisiones tomadas para conseguir el objetivo y las fuentes empleadas para tomar esta decisión.

En el apartado de desarrollo se comenta los pasos realizados para llegar a conseguir los objetivos. Estos pasos se describen de manera cronológica y mostrando el concepto.

En el último apartado que constituiría el proyecto como conjunto, se exponen y explican los resultados obtenidos. Estos resultados son dobles puesto que los objetivos iniciales del proyecto eran dos: el almacenamiento masivo de la información diaria de los servidores del videojuego Travian, y la capacidad de predicción que se consigue con la herramienta desarrollada. Por último, se exponen las conclusiones de este proyecto y se redactan las ideas o perspectivas de futuro que el trabajo puede adquirir.

Palabras clave

Bases de datos, Linux, clasificador, aprendizaje máquina, monitorización, Travian, RaspberryPi.

Agradecimientos

Quería agradecer a toda mi familia por haberme apoyado durante todos los años de carrera que los llevo hablando en esta jerga nuestra tan técnica y en especial este último año que, además de jugar, me he dedicado a desarrollar este proyecto.

Quería agradecer a mis compañeros de facultad, con los que he compartido largas y duras jornadas de estudio en la biblioteca, el apoyo que me han dado en todos los momentos que hemos compartido. En especial a Jorge Sánchez Castillo que no sé cómo habría acabado este año sin él.

Quería agradecer a Aythami el apoyo y la ayuda, así como el empujón para atreverme a proponer esta idea como tfg. A Eduardo por haberme dado la oportunidad de comenzar este tfg.

Y por último quería agradecer a mi compañero "romano" Miguel Ramirez Verdyguer por haber compartido infinidad de horas de este precioso juego y haber plantado la semilla de este proyecto.

INDICE DE CONTENIDOS

INDICE DE FIGURAS

INDICE DE TABLAS

1 Introducción

La industria de los videojuegos está creciendo y evolucionando a velocidades increíbles, generando empleo y moviendo cantidades astronómicas de dinero. En especial, el mundo del videojuego competitivo está captando público a mucha velocidad, lo que implica que grandes corporaciones inviertan dinero en esto con motivos publicitarios. Sin embargo, este terreno es relativamente joven, porque a pesar de llevar muchos años, su auge comenzó apenas unos pocos. Es por esto que muchas de las técnicas que se emplean en sectores más veteranos, apenas se hayan dado a conocer en este ámbito. Una de ellas es el análisis de datos, que está empezando a surgir en el terreno multijugador competitivo, pero que, en otros terrenos de un solo jugador, apenas existe.

1.1 Motivación

Los videojuegos han tenido un crecimiento exponencial desde su creación debido a las posibilidades de innovación que ofrecen, en especial, el mundo de los electronic sports que han tenido una expansión sin igual y han llegado a rivalizar con los deportes más clásicos.

Esta creciente industria de e-sports, que poseen una inmensa cantidad de público, genera unas cifras de datos similares a las que aspiran a tener otros sectores.

Debido a la juventud de este campo, apenas existen herramientas que implementen tecnologías capaces de aprovechar toda esta cantidad de información. Es por esto, que la finalidad de este TFG es generar una herramienta capaz de suplir el vacío que existe ahora mismo en servicios que realicen un uso avanzado de los datos.

1.2 Objetivos

La concepción inicial del TFG es la creación de una herramienta capaz de predecir con antelación si un jugador va a dejar de jugar al videojuego Travian. De la idea anterior se derivan la necesidad de creación de un recolector de datos del juego, un sistema de almacenaje de estos y una herramienta de análisis. El empleo de estas herramientas como servicio “vivo” va a requerir de una profunda instalación y configuración inicial en el sistema operativo GNU/Linux. Va a ser necesario emplear instrumentos que permitan una sincronización de estos servicios para uso compartido de los recursos almacenados. El acceso remoto al servidor va a ser indispensable para el desarrollo del tfg, por lo que será necesarias modificaciones adicionales para permitirlo. Es imprescindible realizar una configuración básica de seguridad para asegurar la integridad de los datos. Se va a programar un sistema de backup por si es necesario restablecer el sistema en caso de fallo. Es imprescindible que todo lo anterior funcione de manera sincronizada, pero sin llegar a estorbarse.

2 Estado del arte

El empleo de técnicas para conseguir un mejor rendimiento en videojuegos es una práctica habitual desde el inicio de estos. En especial el desarrollo de técnicas que pretenden el objetivo de vencer a un rival sea humano o máquina, dan lugar a estudios (11) con la finalidad de encontrar el mejor camino para conseguir este objetivo.

En (11) se pretende si se puede conseguir de manera exitosa ganar al ajedrez empleado ataques de fuerza empleado diferentes estrategias de valoración de posiciones.

(12) Hace uso de un clasificador binario para determinar el resultado en videojuego. Se emplean dos enfoques diferentes desde la perspectiva del jugador y desde la perspectiva del equipo para determinar la condicion de victoria.

Desarrollado sobre el videojuego anterior, (13) hace uso de el algoritmo de regresión logistica para analizar de manera previa al inicio de la partida, quien consiguira la victoria en funcion de los personajes elegidos antes de la misma. Para ello analiza el equipo de manera global, y por cada personaje seleccionado, llegando a tener un 63% de acierto sin conocer de antemano la habilidad individual que posee cada jugador.

En el videojuego Travian, existen multiples herramientas que permiten al jugador un estudio directo de los datos. Estas herramientas realizan un analisis poco desarrollado sobre el objetivo de este proyecto. Esto es debido a que focalizan sus esfuerzos en el desarrollo de diversas funcionalidades adicionales. Estas herramientas son (14) y (15).

2.1 Contexto del videojuego Travian

Travian en un videojuego de navegador que se desarrolla en una época similar a la romana. En la fase inicial, cada jugador elige una de las razas disponibles y se le otorga una aldea que se convertirá en la capital de su imperio. El jugador deberá ampliar los edificios en su aldea, lo que se reflejará en un aumento de la población, y construir ejércitos para conquistar, arrasar o defender aldeas de sus rivales o compañeros. Es un juego en el que las acciones realizadas implican tiempo, por lo que el desarrollo del juego es lento y puramente estratégico. Uno de los factores para conocer la actividad del jugador es el crecimiento de sus aldeas, así como los movimientos de sus ejércitos. El hecho de ser un juego multijugador de ordenador facilita la extracción de datos.

2.2 Herramientas empleadas

2.2.1 GNU/Linux

Es un sistema operativo libre escrito en lenguaje C que deriva de UNIX. Debido al modularidad de su kernel es el S.O. perfecto para emplear en dispositivos de bajas prestaciones. Posee una gran comunidad de personas que hacen posible que disponga de multitud de herramientas y tutoriales para cualquier tipo de funcionalidad.

2.2.2 MariaDB

MariaDB es un gestor de bases de datos que procede del gestor MySQL. Esta creado por el mismo grupo de desarrollo, dirigido por Michael Widenius, debido a la licencia que posee Oracle sobre MySQL. Está ampliamente extendido entre las diferentes distribuciones de Linux.

2.2.3 Python

Python es un lenguaje de programación de alto nivel y multiparadigma. Esto hace que sea el lenguaje de programación más usado en 2019 según el estudio anual realizado por la página stackoverflow. Su gran popularidad ha convergido en el desarrollo de múltiples librerías o módulos que hacen de este lenguaje una herramienta muy flexible. Es por esto por lo que se ha empleado esta herramienta como lenguaje oficial del TFG.

2.2.3.1 Pandas

Pandas es un módulo de Python destinado al manejo de datos y que se caracteriza por su fácil paralelización y eficiencia. También posee una serie de estructuras y funciones internas que facilitan la tarea del programador para describir operaciones sobre los conjuntos de datos. Está escrita como una extensión para el módulo NumPy.

2.2.3.2 Requests

Requests es un módulo de Python que, como dice su lema, pretende una fácil interacción entre el protocolo HTTP y el desarrollador humano. Posee múltiples funciones y estructuras que se encargan de crear una interfaz simple para el programador de manera que pueda evitarse la creación de sockets y demás capas. Esto hace que la interacción entre servidores y clientes web se realice de manera sencilla.

2.2.3.3 Beautiful Soup

Beautiful Soup es un módulo de Python que facilita el análisis y extracción de los datos de documentos HTML. Esta tarea la realiza convirtiendo el documento en un árbol de estructuras y permitiendo al desarrollador navegar por el de manera fácil.

2.2.3.4 MySQL Connector

Este módulo permite la interacción de cualquier programa escrito en Python con una base de datos de tipo SQL. Esto se realiza mediante una interfaz que permite realizar cualquier tipo de acción desde un programa Python sobre la base de datos elegida.

2.2.3.5 Scikit-Learn

Este módulo dispone de todas las herramientas suficientes para realizar sobre un conjunto de datos el aprendizaje maquina deseado. Para este proyecto se han empleado el conjunto de herramientas destinadas al uso de clasificadores.

2.2.4 SBC RaspberryPi 3B+

Los SBCs son dispositivos que se caracterizan por sus bajas dimensiones y capacidad de emplear sistemas operativos. Suelen tener consumos reducidos y disponer de entradas y salidas analógicas debido a su extendido uso como gestor de sensores. Uno de los fabricantes de más renombre debido a su bajo precio y a su licencia open source. Posee una gran comunidad de usuarios que fomentan su desarrollo y facilitan el acceso a estos dispositivos. Se ha empleado el modelo 3B+ de esta casa.

3 Diseño

Como se ha mencionado anteriormente en el objetivo pretendido por el TFG es desarrollar dos herramientas que actuando en conjunto entrenen un algoritmo capaz de clasificar la actividad futura de los jugadores. Estas herramientas realizan su funcionalidad en diferentes pasos mostrados en las siguientes figuras.

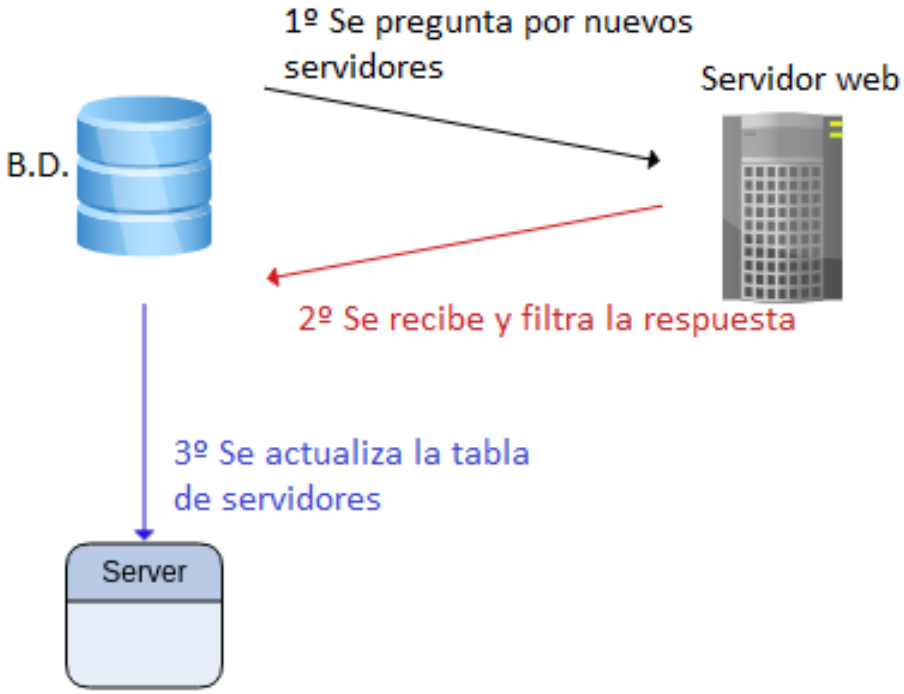

Figura 3.1: Actualización de la tabla de servidores

La figura 3.1 muestra el paso de actualización que realiza la base de datos sobre la tabla de servidores. Este paso es previo a la recopilación diaria de datos de los servidores activos que se muestra en la figura 3.2

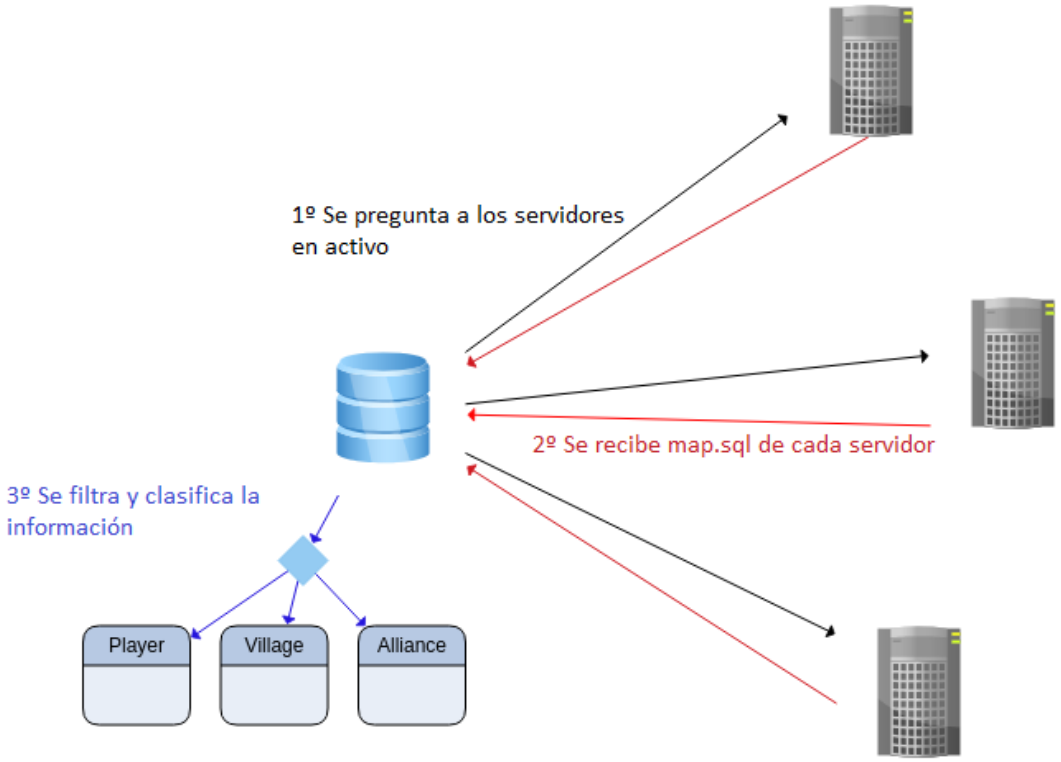

Figura 3.2: El diagrama representa la interacción de la base de datos con los servidores para recoger el objeto map.sql y su almacenamiento

La última parte de la funcionalidad es el empleo de los datos para entrenar un clasificador, para ello deben recorrer un camino de tratado que se refleja en la figura 3.3

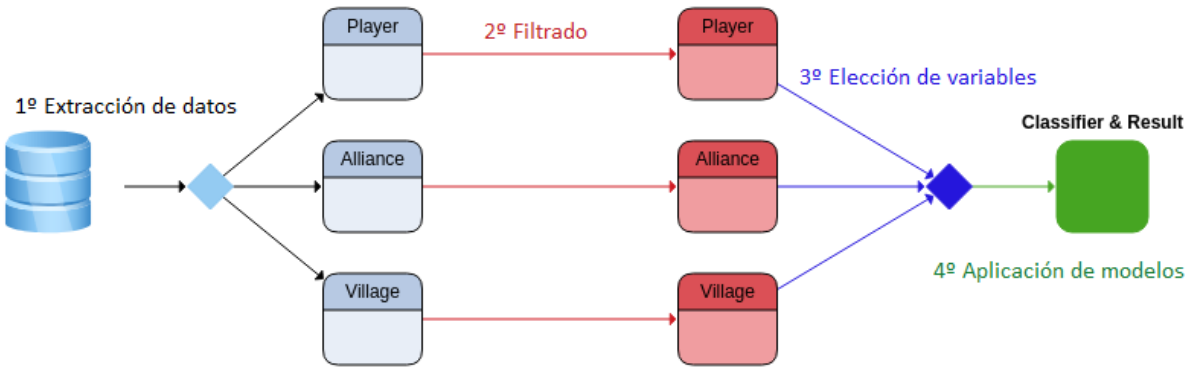

Figura 3.3: Diagrama de camino de empleo de los datos recogidos

3.1 Hardware y S.O.

La adquisición de equipamiento hardware fue la primera planificación que se llevó a cabo en el proyecto. Las decisiones expuestas en este apartado fueron tomadas previamente a la conversión de este proyecto en un trabajo de fin de grado. Por lo que estas adquisiciones estuvieron plenamente condicionadas por el contexto del estudiante y afectaron al resultado final.

3.1.1 Hardware

Ya que el equipo hardware fue enteramente sufragado por el estudiante, era necesario que fuese un equipo económico y que tuviese una capacidad suficiente para poder procesar una cantidad alta de datos, también sería un añadido que generase un volumen de ruido bajo puesto que se esperaba que estuviese encendido de manera permanente.

Los equipos que más se adaptaron a estas necesidades fueron los SBCs o Single Board Computer. Son equipos que por el bajo precio al que se venden ofrecen unas características increíbles en cuanto a las unidades de procesamiento, que llegan a ser quad-core, y el volumen de ruido es prácticamente inexistente. Como precio a pagar, se vio durante el desarrollo que no cumplían las expectativas requeridas en cuanto a su baja memoria ram disponible y sin posibilidad de ampliación.

Me decante por la casa RaspberryPi, que es la que más soporte comunitario ostenta, y por su modelo de placa más reciente la 3B+. A largo plazo se ha visto que fue la mejor decisión puesto que, su escaso Gb de ram ha sido un cuello de botella para la recogida de datos y uso de la base de datos. Como decisión a posteriori, me habría decantado por una placa del fabricante OrangePi ya que muchas de ellas duplican la capacidad ram de la raspberry 3B+. Quiero recalcar que no son equipos pensados para manejar una base de datos de un tamaño medio o grande.

Los SBCs del fabricante RaspberryPi no poseen memoria interna, por lo que es necesario adquirir una memoria externa para instalar el sistema operativo. Se adquirió tanto una tarjeta SD como una memoria USB, empleando la tarjeta SD como memoria de backup y la memoria USB portando el sistema operativo, para así, incrementar el rendimiento de I/Os en el sistema.

3.1.2 S.O. e implementación

La casa RaspberryPi lleva años apostando por una distribución de Linux propia, conocida como Raspbian, que es una distribución basada en Debian adaptada a sus SBCs. Se empleo este sistema operativo en el proyecto dada su popularidad y su inmensa cantidad de tutoriales disponibles. Debido a que es una derivación de Debian y no el propio, existe una limitación en cuanto a paquetes disponibles, esto ha hecho que no se hayan podido llevar a cabo todas las funciones en la placa. Para realizar parte de la configuración inicial se ha empleado este artículo (10)

Puesto que se le quería dar al proyecto un matiz de proyecto profesional, se ha recurrido a diferentes configuraciones imprescindibles como son la monitorización del hardware o la programación de backups que son vitales para la viabilidad de proyectos, así como configuraciones que aportan funcionalidades adicionales. Todas estas serán explicadas en más detalle en la sección de desarrollo. Como fuente de configuraciones en el sistema GNU/Linux se ha empleado la wiki de ArchLinux (1)

3.2 Adquisición de los datos

Sin lugar a duda, encontrar una manera de recoger los datos de manera eficiente, supuso que el número de muestras adquirido sea razonable para poder llevar a cabo un modelo de clasificación y plantearse ofrecer los datos de manera pública, así como plantear este proyecto como un TFG. Puesto que Travian es un juego lento por excelencia, la adquisición de una muestra por día de cada jugador es una monitorización más que aceptable.

3.2.1 Primera Aproximación

Para el primer diseño de recogida de datos se planteó programar un bot al que se le iban a proporcionar una cuenta por cada servidor del que se pretendiesen extraer los datos. El cual, aprovechándose de las técnicas conocidas como web scraping, generaría una sesión en el servidor y, actuando como un jugador, se dedicaría a adquirir uno por uno los datos de cada jugador en el mundo de juego. Además, el número de datos recogidos podía ampliarse ya que se pudría ejecutar una continua recogida de los mismos, así, como poder adquirir otros datos no referidos directamente a los jugadores, como datos de terreno.

Sobre el papel, esta aproximación cumplía ampliamente con las expectativas deseadas de recogida de datos. Se hizo un diseño poco elaborado de un bot que realizase un loggin y recogiese una pequeña muestra de datos y se vio que el tiempo que tardaba en adquirir los datos era de una dimensión no tolerable. Por lo que se desechó la idea.

3.2.2 Segunda Aproximación

La segunda idea para llevar a cabo la adquisición de datos fue la de establecer un canal de comunicación con el soporte del juego en España y ver si se podía llegar a un acuerdo para poder disponer de los datos de unos cuantos servidores. Lamentablemente esta idea no pudo llevarse a cabo debido a la no disponibilidad de los mismo por parte del servicio de soporte.

3.2.3 Aproximación definitiva

La solución definitiva llegó tras una profunda investigación sobre los servicios web que ofrece Travian games y la comunicación con algunas personas que se dedican de manera aficionada a la captación de datos de Travian. La recogida de datos se vale del recurso map.sql que ofrece cada servidor de manera diaria, se explicara más adelante en la sección de desarrollo. El empleo de estas documentaciones ha sido indispensable en el desempeño de esta parte del proyecto (5) (6) (4).

3.3 Base de datos

Uno de los pilares importantes sobre los que se quería orientar este proyecto era la facilidad de acceso al conjunto de datos, ya que se pretende en un futuro hacer este servicio público para todo el que quiera hacer uso de las muestras recogidas. Por este motivo era necesario tener acceso instantáneo a todo el conjunto de muestras y poder seleccionar entre ellas las que deseemos.

Las bases de datos ofrecen una elegante solución al problema previo, ofreciendo una interfaz al usuario en la que parece que todos los datos están "vivos" y permitiendo realizar consultas con diferentes condiciones para solo obtener los datos deseados. Además, con la configuración correcta, se puede otorgar a estás bases de datos un acceso remoto de manera relativamente sencilla que era el objetivo que se buscaba inicialmente.

Me veo obligado a comentar, que la primera aproximación que realicé para almacenar los datos fue en ficheros .csv pero rápidamente, a medida que cogía peso el proyecto, fue una solución que no cumplía las expectativas deseadas.

3.3.1 Elección del gestor

La primera decisión que tuve que tomar nada más decidir que iba a emplear las bases de datos fue la elección del gestor. Puesto que las bases de datos basadas en modelos relacionales son las más extendidas hoy en día y además poseía ligeros conocimientos en la materia, di por sentada esta decisión, así que los gestores debían de ser de bases de datos relacionales.

La decisión fue MariaDB. Es un gestor similar a MySQL y realizado por el mismo grupo de programadores. Poseía conocimientos iniciales en MySQL pero me decanté por MariaDB por ser el paquete oficial de Raspbian y por abogar por el free software, ya que la licencia de MySQL de free software que posee Oracle puede ser revocada en cualquier momento y hacer que este gestor pase inmediatamente a dominio privado. La configuración de todo este gestor ha venido apoyada por su documentación oficial (2).

3.3.2 Diseño relacional

Cuando se afronta el diseño del modelo relacional se suele seguir fielmente las reglas de Codd, o por lo menos las cuatro primeras son imprescindibles para un buen diseño en la base de datos. Estas reglas implican la separación de los datos en tablas y la normalización de estos para evitar redundancias.

Debido a las limitaciones que tenía a nivel de hardware, decidí seguir estas reglas y además intentar reducir al máximo el número de variables que eran combinación de otras, así como de vistas dinámicas que habrían facilitado el análisis posterior de los datos, es decir, intenté realizar un diseño lo más espartano posible. Medida que luego se recompenso ya que uno de los cuellos de botella del proyecto es el manejo del número de muestras frente a la pobre memoria ram del dispositivo.

Como punto para tener en cuenta en este "ahorro" de variables fue que decidí no poner índices propios a cada tabla y en lugar de eso usar una combinación de variables como clave primaria, dando lugar a operaciones de búsqueda menos optimizadas pero que ocupasen menos espacio en memoria. Esta parte la expondré más a fondo en el apartado de desarrollo.

3.4 Gestión remota

Puesto que uno de los objetivos finales de este proyecto es la divulgación de los datos obtenidos en forma de base de datos, es imprescindible que sea un servicio abierto a internet, así como también, el hecho de poder acceder de manera remota a los equipos facilita el desarrollo del proyecto.

3.4.1 Acceso remoto

El acceso a un equipo o servicio desde internet se puede llevar a cabo de múltiples formas y dependiendo de los recursos propios o con participación ajena.

Se había hecho una inversión inicial a nivel de hardware y de tiempo para configurar y hacer funcionar una base de datos en uno equipo propio, por lo que la opción de migrar la base de datos a un servidor como podrían ser los que ofrece Amazon Web Service fue una idea que apenas se barajó.

Para emplear remotamente un equipo del que se realiza un hosting propio se suele recurrir a el empleo de un servidor vpn público o publicar el servicio directamente en internet. La primera solución es la más cómoda ya que la configuración es se realiza en menos equipos y servicios, pero pones en riesgo la privacidad de todos los datos que circulen a través del servidor vpn ajeno, por lo que se implementó la segunda opción. Para publicar un servicio en internet hay que configurar individualmente cada equipo y servicio que se desea exponer a la red externa, además, la configuración en el equipo de salida es crucial para tener un correcto funcionamiento. Se detallarán los cambios realizados en los equipos y servicios en la sección de desarrollo.

3.4.2 Seguridad

La publicación de un servicio directamente en internet supone un riesgo que es necesario securizar ya que él es el propio encargado de lidiar frente a las posibles agresiones externas. Esto es así debido a que el equipo dedicado a hacer de firewall no actúa interponiéndose entre el servicio y esas agresiones.

3.4.2.1 Gestión de usuario y contraseñas

Se ha llevado a cabo una exhaustiva creación de usuarios con privilegios muy acotados a su funcionalidad y un empleo de contraseñas lo suficientemente seguras para evitar que sean rotas por ataques de fuerza bruta. Todos ellos han sido almacenados en gestor de contraseñas multiplataforma como Keepass.

3.4.2.2 IPTables

Aprovechando que se había realizado la configuración sobre un sistema Linux se ha empleado la herramienta del kernel IPTables para realizar un pequeño filtrado de paquetes frente a ataques de fuerza bruta. Los múltiples artículos de la wiki de Arch Linux (1) han sido indispensables para realizar esta configuración.

3.4.2.3 Cambio de configuraciones por defecto

Una de las reglas básicas de la ciberseguridad para exponer servicios en internet, es evitar las configuraciones básicas de los equipos y servicios, ya que esto implica que el agresor sabe de manera sencilla donde debe aunar sus esfuerzos en romper la seguridad. Es por esto que se ha dedicado una parte del tiempo a reconfigurar estas posibles vulnerabilidades en el proyecto.

3.5 Modelo de machine learning

Una de las ideas iniciales para empezar el proyecto era el empleo de los datos para poder estimar el posible abandono de un jugador en el servidor. Decisión que se espera emplear como herramienta en proyectos futuros debido a su enorme potencial dentro del juego. Puesto que el resultado de esta posible decisión es binario, se decidió emplear un clasificador como modelo para estimar. Tanto el filtrado como el enriquecimiento se han podido llevar a cabo gracias a la ayuda de la documentación oficial de Pandas (3).

3.5.1 Filtrado

La funcionalidad del filtrado en esto proyecto consiste en dos simples pasos, marcar a los jugadores que se han quedado inactivos y eliminar sus colas de inactividad. Eliminando las colas de inactividad se consigue tener una muestra limpia del jugador que seguía jugando, pero iba camino de dejar de jugar. De no haber eliminado estas muestras habrían ensuciados los resultados puesto que Travian no elimina a los jugadores inactivos.

3.5.2 Enriquecimiento de variables

El único dato real del que se disponía era el número de habitantes que tenía una aldea cada día, por lo que, para mejorar al clasificador, se han creado variables derivando de la primera para poder modelizar con más precisión a los jugadores.

4 Desarrollo

En esta sección voy a referir en un orden lo más cronológico posible, el desarrollo de las diferentes partes que ha tenido el proyecto, así como las dificultades a las que me he enfrentado durante el mismo. Como excepción, voy a explicar en última instancia la configuración realizada en el S.O.

4.1 Base de datos

Tras ver la posibilidad de una recogida de datos exitosa mediante el servicio maps.sql, se procedió al diseño de la base de datos teniendo en cuenta la información aportada por el recurso, y no la que de verdad existe en el ecosistema Travian.

4.1.1 Map.sql

La aplicación web de Travian ofrece este recurso por cada servidor y es actualizado una vez al día con la información de la situación del día anterior. Para acceder a el mismo, simplemente hay que realizar una petición html al servidor elegido, un ejemplo sería el siguiente:

https://ts20.Travian.com/map.sql

Con esto se iniciaría una descarga de un fichero que contendría la información de la situación de cada aldea el día anterior en el servidor, que sería la siguiente:

Grid: posición en la rejilla que simula el mapa del servidor, tomando como origen la esquina superior izquierda y creciendo hacia la derecha y hacia abajo
X: posición en el eje x
Y: posición en el eje y
Raze: tipo de raza escogida
Id_village: número único por servidor que hace de referencia a la aldea
N_village: nombre que el jugador escoge para su aldea
Id_player: número único por servidor que hace de referencia al jugador
N_player: alias que posee el jugador en el juego
Id_alliance: número único por servidor que hace de referencia a la alianza
N_alliance: alias que posee la alianza en el juego
Population: habitantes que posee la aldea

La sentencia sql iría en este formato:

```
INSERT INTO `x_world` VALUES (1946,141,196,6,23953,'Hyp91',88,'Hig',117,'OnePiece',42);
```

Siendo `x_world` la tabla donde debería ser insertada. Se mencionará este formato en la parte de recogida de datos.

4.1.2 Modelo entidad relación

Para realizar uno diseño relacional y no redundante, el primer paso es hacer el modelo entidad relación.

4.1.2.1 Entidades

La primera parte es definir las entidades y sus atributos. Teniendo en cuenta, que se iba a realizar una única base de datos para almacenar todos los servidores, es necesario incorporar la entidad servidor, así que las entidades quedan definidas así:

Server

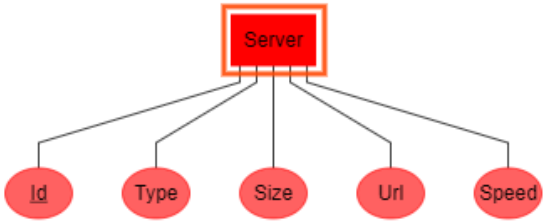

Figura 4.1: Ilustración de la entidad fuerte Server

Esta entidad hace de entidad bisagra ya que es la entidad fuerte de todas las demás entidades que forman un servidor, como atributos posee su número identificador, Id, que hace de clave primaria, Type, que indica el modo de juego del servidor, Size, que contiene el tamaño del mapa donde se juega, Speed, que es el ritmo de juego y por último la dirección url donde tendrá que apuntar el recolector de datos.

Time

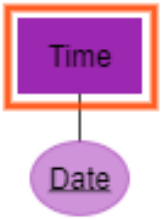

Figura 4.2: Entidad referente al tiempo

La entidad Time, junto con su atributo Date, que es su clave primaria, es la encargada de mantener la coordinación temporal del resto de entidades.

Player

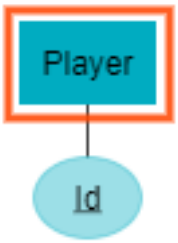

Figura 4.3: Representación de la entidad jugador

El unico atributo de jugador es su id ay que el resto de varibles que aparentemente debería poseer son dependientes de la relación con las otras entidades, por lo que se reflejaran como atributos de relación.

Alliance

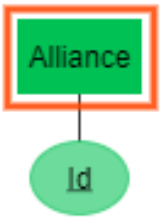

Figura 4.4: Representación de la entidad alianza

Al igual que ocurre con Player, Alliance contextualiza sus atributos cuando se da la relación.

Village

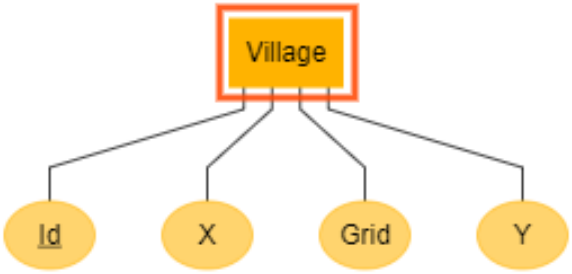

Figura 4.5: Representación de la entidad aldea

La última entidad es Village, que posee su Id como clave primaria y X, Y y Grid como atributos de posecionamiento en el mapa, estos ultimos no pertenecen a la clave primaria, a pesar de ser únicos, debido a que se puede destruir una aldea y fundar otra en el mismo lugar.

4.1.2.2 Relaciones

Muchas de las relaciones son relaciones triples, hecho muy poco recomendado por los diseñadores de modelos y que implica decisiones un poco opacas a la hora de dibujar el modelo. Esto ha sido debido al factor tiempo del que dependen las relaciones ya que este es un factor imprescindible. Se han representado los elementos débiles como piezas de puzle.

Village-Server

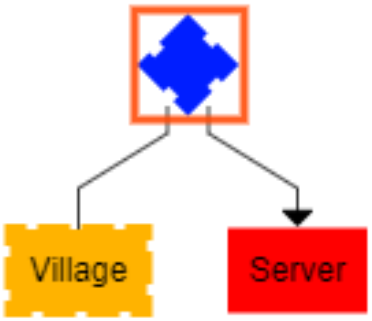

Figura 4.6: Representación de la relación de dependencia

Relación que refleja la pertenecia de una aldea a un servidor. Decir que Village es una entidad debil con respecto a Server, por lo que su Id pasa a ser un discriminante y el Id de Server pasa a formar parte de su clave primaria.

Player-Server y Alliance-Server

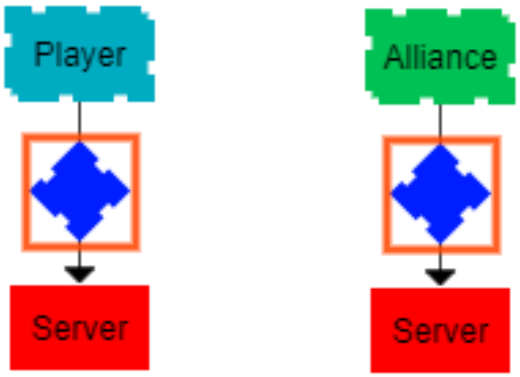

Figura 4.7: Dos entidades débiles con respecto a la fuerte

Una de las decisiónes para reducir el consumo y rellenar tablas ha sido en la creación de estas dos relaciónes. Cada una de estas entidades debería tener otra relación que dependiese de la entidad fecha y poseer los atributos de nombre y algunos atributos extra como el número de aldeas por jugador o el número de jugadores, dando lugar a dos tablas que reflejasen el estado actual de las dos entidades. Esta reducción de relaciones y atributos ha dado lugar a que se haya tenido que dedicar más esfuerzo y tiempo en la parte de filtrado de datos. Decisión que será una de las candidatas a ser revocadas en un futuro. Decir que la clave primaria de Player y Alliance pasa a estar formada por la clave primaria de Server.

Village-Player-Time

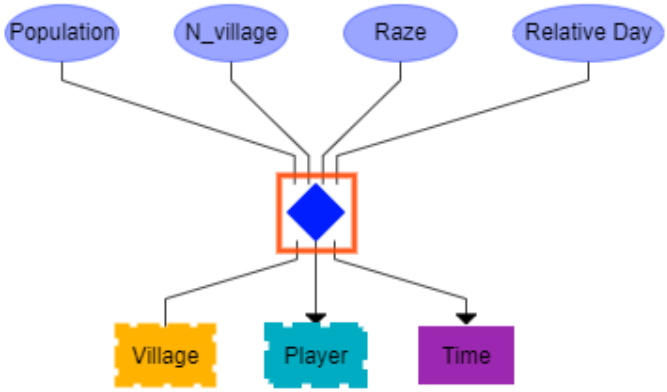

Figura 4.8: Relación que dará lugar a la tabla más relevante de la base de datos y evita la creación de dos tablas alternativas de village y player con time.

Es en esta relacion donde se ve el estado de cada jugador con sus aldeas, la tribu a la que pertenece la aldea y el nombre de la misma. Esta relación es entre entidades de igual peso por lo que las claves primarias se mantienen. El empleo de pocos campos ha hecho que se ahorre en memoria pero se incremente el tiempo de calculo. Relative day se puede calcular a posteriorí pero es un atributo los suficientemente importante como para haberlo considerado. Por ejemplo, no se ha añadido el tiempo relativo de juego de la aldea, que es menos importante, pero támbien se emplea en calculos futuros.

Player-Alliance-Time

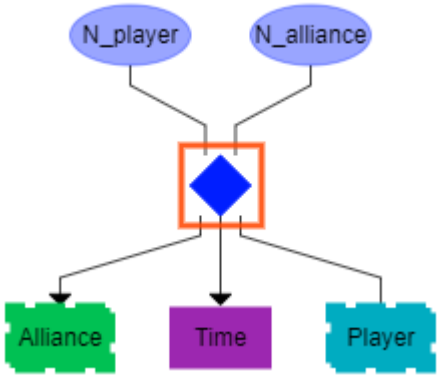

Figura 4.9: Esta relación evita la creación de la tabla Alliance-time

Esta relación define la pertenencia entre jugadores y alianza, asi como los nombres de cada uno, esto hace que el nombre de alianza sea un elemento redudundante, pero nos ahorramos la inserción de elementos en otra tabla.

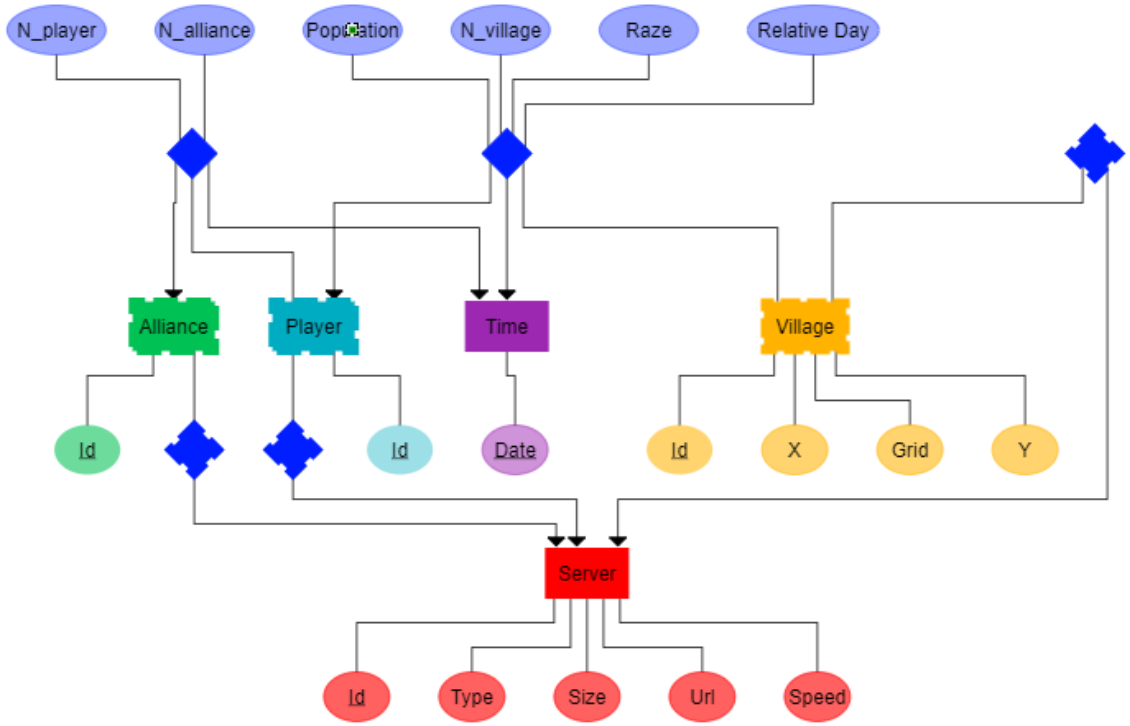

Figura 4.10: Diagrama Entidad-Relación completo

4.1.3 Modelo relacional

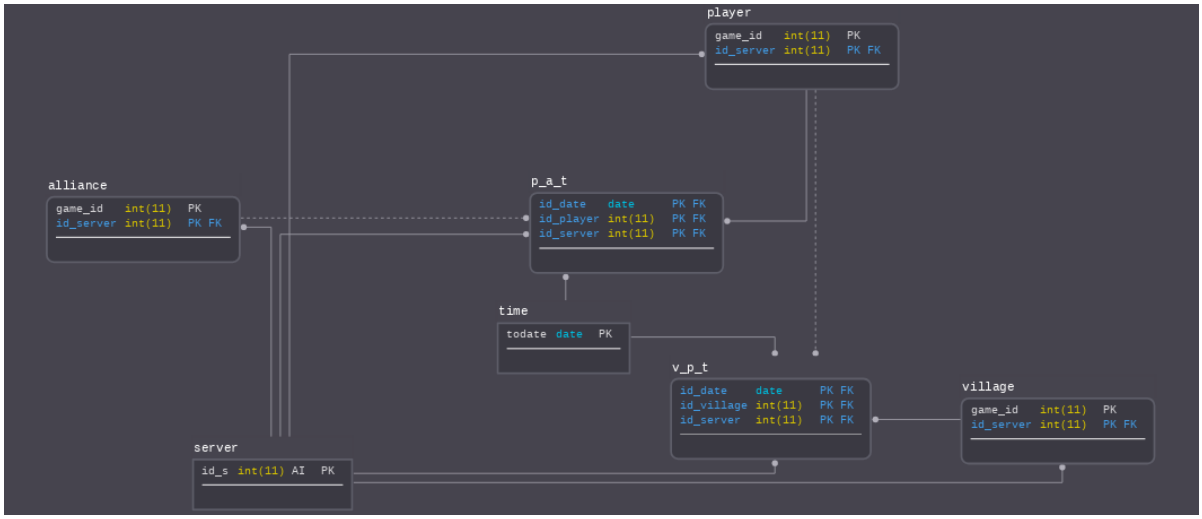

Figura 4.11: Representación final de las tablas almacenadas en la base de datos

Siguiendo las reglas de transformación de modelo entidad relación a modelo relacional, obtenemos una serie de tablas que van a ser donde van a estar almacenados los datos. Estas tablas salvaguardan su integridad no repitiendo su una clave primaria en ninguna de las

entradas. Las líneas que acaban en punto quieren decir que transfiere la clave primera desde donde se origina la línea a donde acaba el punto formando parte de la clave primaria de la tabla, mientras que las líneas de rayas solo realizan un traspaso de la clave primaria, pero se puede repetir en la tabla destino.

Como se ha comentado previamente, se ha conseguido eliminar la necesidad de crear dos tablas extra de la situación temporal en el servidor de las alianzas y jugadores resumiéndolo en p_a_t. También se ha evitado repetir la clave primaria del servidor varias veces en las tablas v_p_t y p_a_t.

4.2 Adquisición de datos

Es en este punto donde se decidió el empleo de Python3 como lenguaje oficial del proyecto. La cantidad de librerías o módulos, como son conocidos dentro del lenguaje, permiten una muy fácil interacción con bases de datos de tipo SQL, además de múltiples funcionalidades.

4.2.1 Web scrapping

La fase de extracción de datos se realiza en dos simples pasos, el primero actualiza la base de datos de los servidores activos y el segundo va servidor por servidor pidiendo el recurso map.sql. Estos dos pasos se valen de interactuar con servidores web y paginas html, que están pensadas para uso humano, de manera automática. Esta práctica conocida como web scrapping limita con la ética de uso de estos servicios, por lo que hay administrador@s web que emplean técnicas para intentar impedir este uso.
Se han tomado precauciones para evitar un posible bloque por parte de estos servicios.

- Se ha camuflado al scrapper como si fuese un usuario normal navegando con el navegador Mozilla. Esto se consigue generando una sesión html que almacene cualquier tipo de cookie y en cada conexión del servidor aportando unos datos falsos tales como un tamaño de pantalla, o comunicando el user-agent que se está empleando.
- Se ha evitado un uso excesivo de la red. El empleo de un uso abusivo del servicio web puede delatarlo como un web scrapper y se han empleado medidas que reduzcan las peticiones en caso de saturar el servidor.

4.2.2 Extracción de servidores activos

Realizando una consulta a la página web oficial de servidores de Travian () y parseando esta información, se consigue una ristra de servidores en activo que posteriormente se emplea para actualizar la tabla de servidores de la base de datos.

4.2.3 Extracción de la información de cada servidor

Tras la actualización de la tabla de servidores, se va realizando una consulta propia a cada servidor, viendo su disponibilidad y pidiendo el recurso map.sql

4.2.4 Parseo de map.sql

Una vez obtenido el recurso map.sql de un servidor, se parsean los datos disponibles. He de decir en este punto, que a pesar de que Travian parece facilitarnos el trabajo ofreciéndonos esta información en forma de sentencias SQL, se nota que no existe un desarrollo de modelo entidad-relación en su base de datos, o por lo menos no quieren compartirlo. Esto es así porque debido al formato en el que ofrecen las sentencias, parece que nos invitan a juntar toda la información del servidor en una tabla, cosa totalmente inóptima. Hecho que hace que las supuestas sentencias favorables sql, sean un incordio para el desarrollador.

4.2.5 Inserción en la base de datos

Con los datos obtenidos en forma de variables y usando el módulo de SQL-connector, la inserción en las diferentes tablas se realiza de manera relativamente sencilla. Estas inserciones se pueden llegar a optimizar en gran medida empleando prepared staments. Hay que tener en cuenta las excepciones causadas por la inserción de un dato repetido ya en la base de datos.

4.3 Gestión remota

Como se ha comentado en diseño, se optó por publicar los servicios directamente desde la propia red local. Esta red local posee una salida a internet con una ip pública y dinámica, por lo que fue necesaria la creación de un script para conocerla ya que varía con el tiempo, se detallara en S.O.

4.3.1 Configuración en el router

El primer paso necesario para comunicar un equipo con el exterior es saber identificarlo, por lo que a los equipos implicados en todo el TFG, se les ha omitido del servidor dhcp y se les ha otorgado una ip estática de la red interna. Esta configuración es sencilla, únicamente hay que acceder al router y asignar manualmente ips estáticas según la dirección mac de la tarjeta de red de los equipos.

La siguiente configuración ya requiere de conocimientos más avanzados y es la creación de vips con los servicios correspondientes. Se han creado dos, una para el servidor ssh de la raspberry y otra para la base de datos. Las vips permiten a las conexiones entrantes realizar un baipás al firewall para comunicarse al equipo asignado en el puerto asignado.

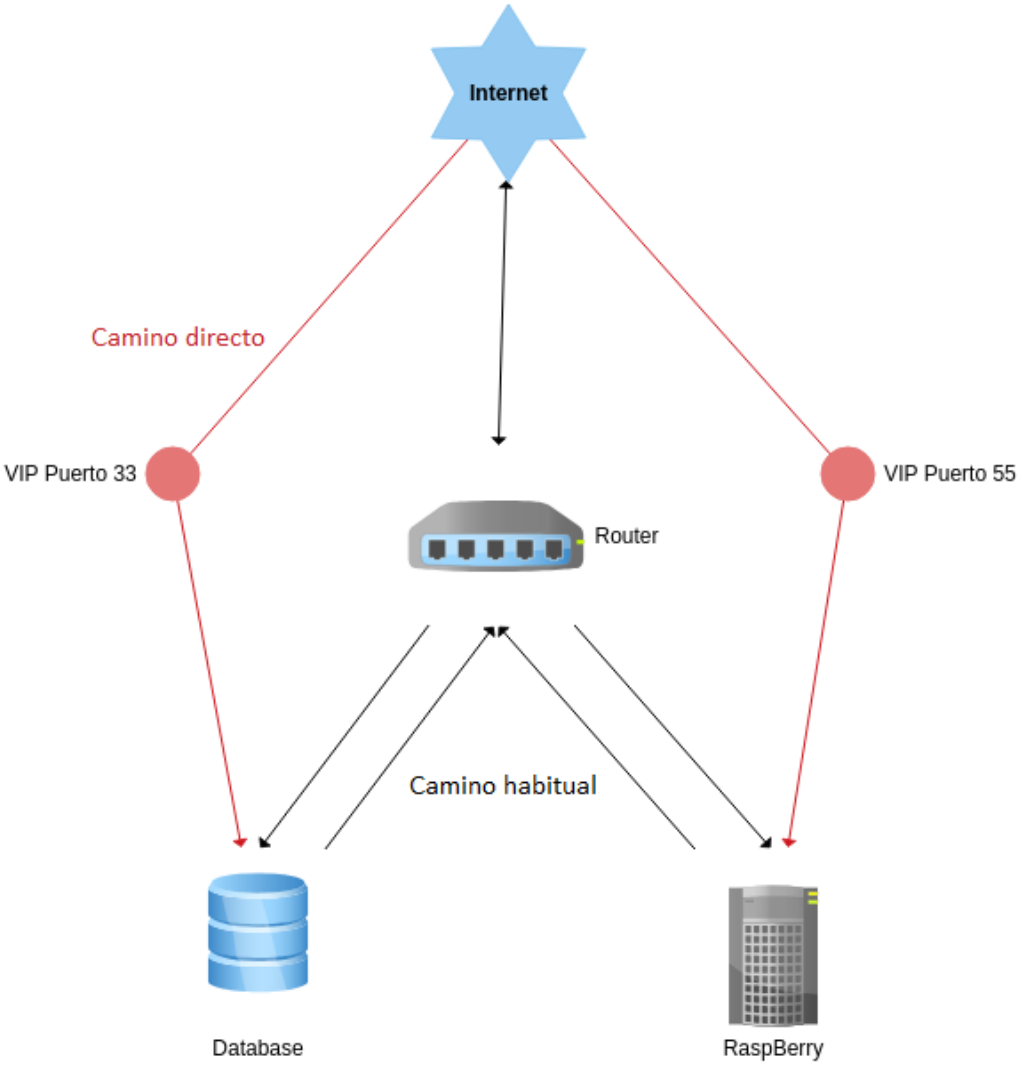

Figura 4.12: Función del empleo de vips en el router y como evitan interactuar con el mismo

Las vips en color rojo, son un acceso sin control de direcciones ip.

4.3.2 Cambio en las ip de escucha

Por defecto, los servicios que operan dentro de un sistema están enlazados con ip de loopback. Esto hace que no pueda existir una comunicación desde el exterior del equipo con ese servicio. Ha sido necesario acceder a los ficheros de configuración de la base de datos de MaríaDB y cambiar la ip de loopback por la ip 0.0.0.0.

4.3.3 Seguridad

La publicación de un servicio en internet hace que el propio servicio sea el responsable de su propia securización. Por lo que una configuración por defecto no es suficiente.

4.3.3.1 Control de usuarios y claves

Se ha hecho creado un conjunto de usuario con un acceso extremadamente restringido a los recursos necesarios para llevar a cabo su función y a la situación desde la que acceden al servicio.
SSH cuenta con un solo posible usuario con el que se ha realizado un intercambio de claves pública y privada en un lugar seguro. Estas claves se envían de manera cifrada para aumentar la seguridad, es decir, es necesario conocer dos claves diferentes y para poder acceder al equipo por SSH.
La base de datos dispone de tres usuarios en total, uno para la selección de filas en determinadas tablas desde el mismo equipo, otro para la inserción de filas en determinadas tablas desde el mismo equipo y otro para la selección de columnas con acceso remoto. Existe un cuarto usuario que dispone de todos los privilegios, pero es necesario un acceso local y poseer derechos de super usuario Linux.
Por supuesto todas las claves poseen un mínimo de 20 caracteres alfanuméricos.
La creación y configuración de los usuarios de la base de datos ha sido un pequeño desafio puesto que posee una nomenclatura totalmente diferente a otros servicios. Además, el hecho de que se puedan restringir el acceso mediante el protocolo ip en la propia configuración del usuario o en el apartado de creación, me ha supuesto el empleo de horas de intentar resolver este problema.

4.3.3.2 Cambio de puertos predeterminados

Se han cambiado los puertos por defectos de ambos servicios por unos puertos del rango de uso privado para más seguridad. Esto se ha realizado modificando los ficheros de configuración.

4.3.3.3 IpTables

Usando este módulo del kernel de Linux se ha configurado un pequeño firewall que bloquea el acceso al equipo de manera temporal a cualquier ip que realice una autenticación errónea en cualquiera de los servicios más de tres veces. Esto evita el empleo masivo de ataques de fuerza.

4.4 Modelo de machine learning

4.4.1 Filtrado

Este punto ha supuesto sin lugar a duda el mayor reto de todo el TFG. La dificultad ha sobrevenido de mi falta de practica con la nomenclatura de Python y el módulo Pandas, el desconocimiento con el que decidí usar este módulo, esto implica un desconocimiento en estructuras, funciones y el concepto con el que se ha desarrollado, y la unión del paradigma de la programación funcional con el manejo de matrices de datos.
Esto ha implicado que los diferentes resultados de los clasificadores se vean empobrecidos, pues como ya diré en el apartado de conclusión, la falta de tiempo ha hecho que el enriquecimiento de variables y las diferentes opciones de los clasificadores, así como la

selección de los servidores idóneos, haya sido toda realizada en una cantidad de tiempo no correspondiente.

Seguidamente voy a describir el algoritmo por el que he obtenido las ristras temporales de actividad de los jugadores marcadas y limpias:

Tras la importación de las filas de la base de datos como

Id_player	Id_village	Population	Tribe	Relative_day

Tabla 4.1: Estilo de inserción

Se procede a su ordenación por Id_player, Id_village, Relative_day. Posteriormente, se genera una columna adicional, 'Grow', agrupando las muestras por jugador y aldea y calculando la diferencia de poblaciones en los diferentes días.

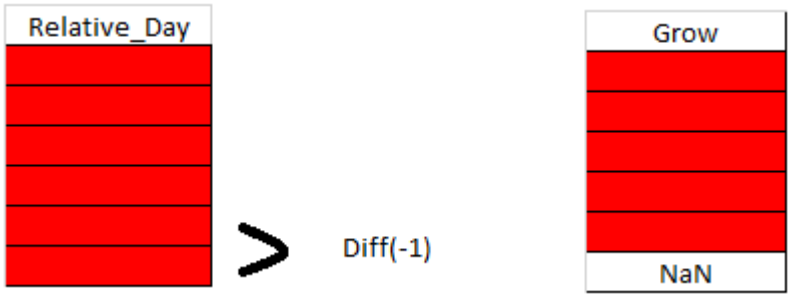

Figura 4.13: Calculo de la diferencia con la fila anterior a nivel matricial

Tras este paso, se vuelven a agrupar las muestras por jugador y día relativo, esto quiere decir, que se generan subgrupos del estado de todas las aldeas por día y jugador, y se suma el crecimiento de las aldeas, si es existe alguna aldea con un crecimiento superior a 0, se considera que ese jugador ha tenido actividad ese día. Con esta información se genera un nuevo DataFrame que posee una columna con los días de actividad de cada jugador.

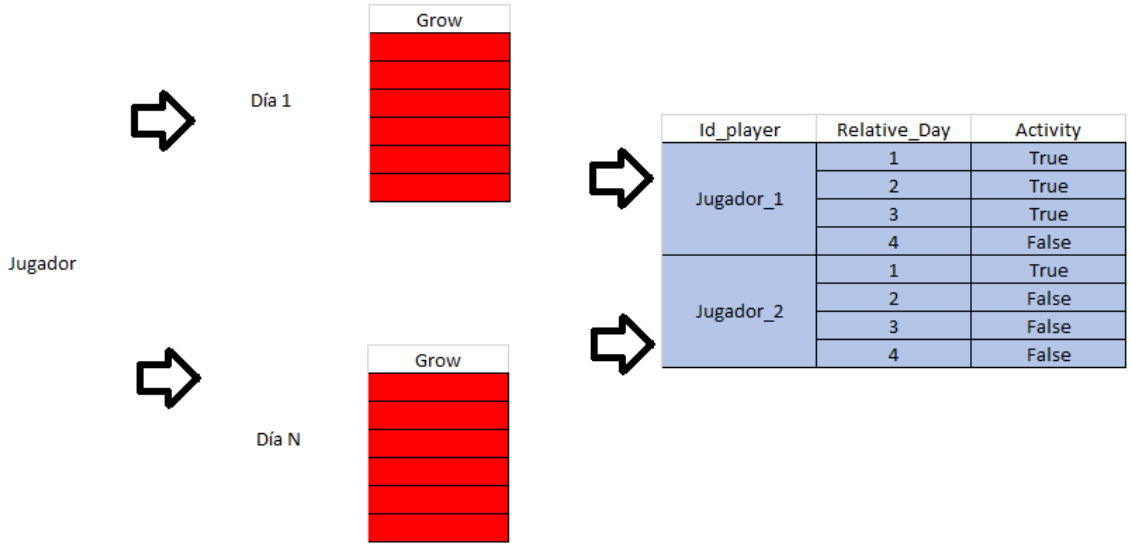

Figura 4.14: Operación de groupbay-apply-junction

Para determinar si un jugador se ha quedado inactivo se aplica una técnica de suma en ventana, esto quiere decir que se suman los días dentro de la ventana y si resulta que el valor es mayor que 0, ha habido un día de actividad por lo que en ese fragmento no resulta inactivo. El número de muestras que emplea la ventana es el número de días que consideramos que un jugador necesita de inactividad para ser considerado inactivo.

Figura 4.15: Operaciones realizadas sobre la ventana obteniendo un escalar

Aplicando esta operación sobre cada jugador se consigue marcar al conjunto de jugadores inactivos. Pasado este punto, ya solo queda eliminar de la muestra de jugadores inactivos, los días posteriores al día en el que se marcan como inactivos. Esto último se realizaría aplicando una función de filtrado a cada jugador-aldea de manera individual y no de manera matricial, por lo que el rendimiento en este punto es menor.

Id_player	Relative_Day	Activity	Inactivity
Jugador_1	1	True	False
	2	True	False
	3	False	True
	4	False	NaN
	5	False	NaN

Id_player	Id_village	Relative_Day
Jugador_1	Aldea_1	1
		2
		3
		4
		5
	Aldea_2	2
		3
		4
		5

Id_player	Id_village	Relative_Day
Jugador_1	Aldea_1	1
		2
	Aldea_2	2

Figura 4.16: Eliminación de colas

4.4.2 Enriquecimiento de variables

Puesto que aparentemente solo contábamos con la variable población, no es posible realizar un clasificador asignando como única variable la población, es más, deberíamos tener el mismo número de muestras de cada jugador para que esto se pudiese llevar a cabo. Por lo que se ha optado por generar una ristra de variables que modelicen el comportamiento de la variable población y que tuviesen en cuenta su variación temporal, como puede ser el crecimiento, su desviación, etc. En este primer enfoque se ha tenido solo en cuenta los valores absolutos del jugador, y no la diferente variación de sus aldeas.
Para realizar esta tarea se ha vuelto a emplear el módulo pandas con su utilidad de aplicar funciones mediante diccionarios en los DataFrames.

4.4.3 Aplicación de modelos

Tras el filtrado de datos y el enriquecimiento de variables, se tiene como resultado dos matrices en la que una contiene las variables elegidas de cada jugador y otra si el propio jugador se considera inactivo o no. Para el entrenamiento y prueba de los modelos se lleva a cabo una separación de estas matrices en submatrices de prueba y de entrenamiento. Es importante que las muestras de la matriz de test estén equilibradas en relación de 50%. El número de muestras debe ser igual para tener un resultado de precisión con respecto al resultado que obtendríamos decidiendo al azar. De no ser así, el resultado podría no ser entendible.

4.5 S.O.

Este último apartado ha sido desarrollado durante todo el proceso que ha durado este proyecto y es por eso por lo que es importante conocer los desarrollos de los anteriores apartados para comprender este.

4.5.1 Monitorización

Como parte importante de un proyecto software es la vigilancia de los equipos hardware, en este proyecto se han realizado una serie de scripts en bash de monitorización para controlar las temperaturas, cargas de trabajo, memoria ram en uso, memoria de almacenamiento empleada y voltaje consumido por parte de los procesadores de la raspberry. Esta información se almacena de manera continuada en una base de datos SQL de poco consumo como es SQLite.

4.5.2 BackUps

Se ha asegurado la integridad de los datos y diversas configuraciones realizando de manera programada el respaldo de la información. Se han empleado dos dispositivos para realizar esta función, el primero ha sido una tarjeta SD integrada en la propia raspberry y el segundo ha sido un dispositivo de almacenamiento remoto. En ambos casos se ha tenido que programar su montaje en el sistema.

4.5.3 Automatización

En la mayoría de los procesos se ha empleado el daemon cron como mecanismo de automatización haciendo uso de la interfaz crontab. En los casos en los que implicaba algunos requisitos más concretos, se ha servido de la definición de servicio Linux.

4.5.4 Cliente SMTP

Se ha realizado la instalación y configuración de un cliente de correo SMTP, así como la sincronización de este mismo con una cuenta de correo. La motivación por la que se ha llevado a cabo este elemento consta de dos puntos. El primero ha sido la necesidad de conocer la ip actual de la que dispone mi salida a internet para poder hacer funcionar la configuración remotos, esto se ha hecho mediante un simple script que reenvía la ip en caso de modificación de la misma. El segundo punto ha sido debido a la necesidad de controlar el funcionamiento del daemon cron.

5 Integración, pruebas y resultados

En este apartado se pretenden mostrar los resultados obtenidos por las herramientas desarrolladas. Como se pretendía el desarrollo de dos servicios, los resultados que se mostrarán serán el número de entradas que almacena la base de datos y los resultados obtenidos por el clasificador.

5.1 Base de datos

En la base de datos se ha almacenado los datos capturados durante un periodo de 30 días. Las capturas diarias son una aproximación a las realizadas. Puesto que los servidores tienen un periodo de duración de en torno a 300 días, por lo que prácticamente se capturo la integridad de los servidores el primer día.

Elemento	Servidores	Alianzas	Jugadores	Aldeas	Estado aldeas	Estado jugadores	Espacio de almacenamiento(Gby)
Total	231	32158	384850	1387310	29.060.079	7.292.956	7,2
Diario	X	1072	12828	46244	968770	243098	0,24

Tabla 5.1: Número de entradas almacenadas

5.2 Clasificador

Para el clasificador se han empleado diferentes modelos ampliamente implementados y que tienen un fácil acceso desde la biblioteca de Python (Sklearn Documentation, s.f.). Para realizar el entrenamiento se han empleado varios servidores con características similares.

Modelo empleado	Regresión logistica	Maquinas de soporte vectorial	Clasificador Bayesiano	Arboles de decisión
Precisión alcanzada	79,95%	65,61%	31,72%	11,06%

Tabla 5.2: Resultados de predicción

El enriquecimiento de variables es un punto clave para amentar la precision de estos algoritmos por que en ensayos futuros se realizara un estudio más a fondo en ese aspecto para mejorar las precisiones.

6 Conclusiones y trabajo futuro

6.1 Conclusiones

A lo largo de todo el proyecto se ha tenido que lidiar con diferentes tipos de problemas, lo que ha llevado a tomar una serie de decisiones y obtener una serie de conclusiones que se refieren a continuación. Estas conclusiones se catalogan en tres puntos que son las tres ideas con las que se concluye el proyecto:

La primera idea es la dificultad que se encuentra al afrontar un proyecto. Como proyecto se entiende a la entidad que nace como una idea y se pretende desarrollar con un tiempo y un presupuesto limitados, y es importante recalcar que son limitados puesto que ha condicionado de manera muy relevante las decisiones tomadas. Existen multitud de elementos que requeriría poco tiempo de realizar, que desarrollados podrían mejorar enormemente el rendimiento del proyecto, y que por falta de tiempo no se han podido apenas plantear. También es importante mencionar la dificultad de arreglar planteamientos iniciales erróneos.

La segunda idea conseguida es la relevancia en cualquier ámbito del rol del analista de datos, en especial en el mundo de los videojuegos. Es verdad que, en los videojuegos más punteros, los equipos suelen tener una persona dedicada a este trabajo, pero ni se le da la relevancia que debe, ni se dedica suficiente presupuesto como para tener un impacto clave en el desarrollo de estos. Uno de los dilemas a los que se enfrentan los videojuegos online hoy en día, es al hecho de que hay jugadores que realizan trampas para ganar, pues estudiando los datos que generan estos jugadores, se podría realizar una herramienta que intentase desenmascararlos, hecho que no se lleva a la práctica en casi ningún videojuego.

La tercera idea trata sobre la ética de los de la capacidad de predicción y actuación que obtienen los algoritmos de aprendizaje automático y las inteligencias artificiales. En el ámbito de diseño de videojuegos, la dificultad de una mecánica puede estar pensada por el propio diseñador para transmitir la experiencia que desea al jugador, cosa que, empleando métodos de ML e IA puede quedar totalmente destruida por su uso. En el videojuego Travian, existe una mecánica que consiste en atacar, empleando este TFG podríamos sacar exceso partido de esta misma ya que no está pensada para ser explotada. Es una herramienta de doble filo que aplicada a la vida real puede dar lugar a situaciones que no están contempladas creando un desequilibrio entre las personas.

6.2 Trabajo futuro

Dado que el TFG realizado ha sido un proyecto propuesto por el estudiante, las expectativas son amplias.

Como objetivos a corto plazo, se pretende realizar una completa paralelización del mecanismo de extracción, así como el de filtrado de datos, realizando todo ello de manera programada.
Para llevar a cabo esta tarea, se pretende adquirir un conjunto de SBC de bajo coste para montar un cluster tipo Beuwolf que junto con modificaciones en el código dará lugar a altos rendimientos en el tratado de los datos.

También se pretende a corto plazo aumentar el número de variables enriquecidas y entrenar modelos en diferentes periodos de tiempo para intentar mejorar los resultados obtenidos.

Como objetivo a medio plazo se espera poder monitorizar una mecánica del juego conocida como “vaquear” que consiste en atacar a gente inactiva. Con los datos obtenidos se pretende entrenar un modelo que permita exprimir al máximo esta tarea.

Como objetivo final del proyecto, se espera desarrollar una interfaz sobre la herramienta de mensajería Telegram, que, haciendo uso de sus bots, permita al usuario gestionar su perfil de jugador, así como realizar funciones adicionales, de manera similar a si estuviese escribiendo un mensaje.

También se está barajando realizar una herramienta web para el buscado de jugadores con posibilidad de quedarse inactivos.

Referencias

1. Arch Wiki. [En línea]
https://wiki.archlinux.org/.

2. MariaDB Documentation. [En línea]
https://mariadb.com/kb/en/library/documentation/.

3. Pandas Documentation. [En línea]
http://pandas.pydata.org/pandas-docs/stable/.

4. Beautiful Soup Documentation. [En línea]
https://www.crummy.com/software/BeautifulSoup/bs4/doc/.

5. Requests. [En línea]
https://2.python-requests.org/en/master/.

6. Mysql-Connector Documentation. [En línea]
https://dev.mysql.com/doc/connector-python/en/.

7. Sklearn Documentation. [En línea]
https://scikit-learn.org/stable/documentation.html.

8. The Python Tutorial. [En línea]
https://docs.python.org/3/tutorial/.

9. Carbonell, Lorenzo. Gestionar bases de datos. [En línea] 24 de Noviembre de 2017.
https://www.atareao.es/tutorial/raspberry-pi-primeros-pasos/gestionar-bases-de-datos/.

10. Carbonell, Lorenzo. Volando con la Raspberry desde USB. [En línea] 21 de Noviembre de 2018.
https://www.atareao.es/tutorial/raspberry-pi-primeros-pasos/volando-con-la-raspberry-desde-usb/.

11. Programming a Computer for Playing Chess. SHANNON, CLAUDE E. 1949, Philosophical Magazine.

12. N Kinkade, L Jolla, K Lim. DOTA 2 Win Prediction. 2015.

13. Atish Agarwala, Michael Pearce. Learning Dota 2 Team Compositions. 2014.

14. Getter-Tools. https://www.gettertools.com/es/. [En línea]

15. Travibot. https://travibot.com. [En línea]

Glosario

API	Application Programming Interface
Ip loopback	Dirección que apunto al propio equipo
DHCP	Dynamic Host Configuration Protocol Protocolo que asigna de manera automática direcciones IP
VIP	Virtual IP Simula la posesión de una nueva IP asignando al servicio un puerto especifico
SSH	Secure Shell Protocolo de acceso a un equipo de manera cifrada
Prepared stamen	Sentencia precompilada en una base de datos
SMTP	Simple Mail Transfer Protocol
Cluster	Conjunto de equipos que actúa de manera sincronizada
Daemon	Es un proceso cuya función no es interactuar con el usuario
BackUp	Copia de seguridad
Mac	Dirección física de una tarjeta de red
IP	Dirección dentro de una red
Router	Dispositivo cuya funcionalidad es la conexión entre equipos y redes
Parsear	Extraer la información de un documento empleando herramientas digitales